TECHNISCHE UNIVERSITÄT
CAROLO-WILHELMINA
ZU BRAUNSCHWEIG

UNIVERSITÄTSBIBLIOTHEK
BRAUNSCHWEIG

SPAR: Service-based Personal Activity Recognition for Mobile Phones

Martin Berchtold[1], Matthias Budde[2], Dawud Gordon[2], Hedda Schmidtke[2] and Michael Beigl[2]

[1]Braunschweig: Institute of Operating Systems and Computer Networks (IBR)
[2]Karlsruhe: Institute of Telematics, Pervasive Computing Chair, Karlsruhe Institute of Technology (KIT)

Published: 31.05.2010

http://www.digibib.tu-bs.de/?docid=00033585

http://www.digibib.tu-bs.de/?docid=00033585 31/05/2010

SPAR: Service-based Personal Activity Recognition for Mobile Phones

Martin Berchtold[1], Matthias Budde[2], Dawud Gordon[2], Hedda Schmidtke[2] and Michael Beigl[2]

[1]Institute of Operating Systems and Computer Networks (IBR), TU Braunschweig
[2]Institute of Telematics, Pervasive Computing Chair, Karlsruhe Institute of Technology (KIT)

May 31, 2010

Abstract

Smart phones have become powerful platforms for mobile communication and applications. This paper presents basic technology that will enable the phone to extend such applications with context awareness under realistic conditions. Recognition is carried out by a service-based context recognition architecture which creates an evolving classification system based on feedback from the user community. The approach uses classifiers based on fuzzy inference systems which use live annotation to personalize the classifier instance on the device to the its user. Our recognition system is designed for everyday use: it is independent of placement (no assumed or fixed position), requires only very little (1-3 minutes per activity) personalization effort from the user and can detect a high number of activities. The results demonstrate the ability of the system to use personalization and the user community as forces for optimization, achieving classification rates upwards of 97% for 10 classes in an evaluation with 20 users and over 500 minutes of data.

1 Introduction

Although smart phone devices are powerful tools, they are still passive communication enablers rather than active assistance devices from the point of view of the user. The next step is to introduce *intelligence* into these platforms to allow them to proactively assist users in their everyday activities. One method of accomplishing this is by integrating situational awareness and context recognition into these devices. Smart phones represent an attractive platform for activity recognition, providing built-in sensors and powerful processing units; they are capable of detecting complex everyday activities of the user (e.g. standing, waking, biking) or the device (e.g. calling), and they are able to exchange information with other devices and systems using a large variety of data communication channels.

Figure 1: Some activities recognized with the phone.

Several approaches to smart phone based recognition (e.g. [17, 18, 4, 10, 16]) have been published which demonstrate the importance of research in this field. The architecture presented here builds on this research, but has several advantages over previous approaches, thus enabling context awareness under realistic conditions. Orientation Independence: one problem in mobile

http://www.digibib.tu-bs.de/?docid=00033585 31/05/2010

context recognition is that the way in which the device is carried greatly affects the ability of conventional classifiers to recognize activities. The classifier structure presented here overcomes this obstacle using a novel training structure. Class Diversity: The SPAR approach provides a platform for classification of many different activities simultaneously (here 10 classes). Usability: The SPAR system does not require prior knowledge or abilities on the part of the user in order to operate the system, and user mistakes will not worsen performance. Online Personalized Optimization: The system uses a fun annotation program on the mobile device to gather new training data which is used to improve the recognition algorithm *at run-time*. Crowd-Sourcing: Not only does this feedback optimize local recognition, but it also is used to improve performance over the whole community of application users. This also means that feedback from remote users in the community improves recognition for local algorithms, which represents a *novel break* from conventional recognition algorithms.

To use the SPAR system, the user downloads a base recognition app for the mobile phone that already provides decent recognition of everyday activities (see Fig. 1) and can be used immediately. The user is then able to improve recognition rates by performing a few personal training steps, providing feedback to the system. The system does not apply any further restrictions, meaning any learning curve should be avoided and normal usage and carrying of the device should not be affected.

1.1 Related Work

Table 1 contains an overview of related work published on the topic of personalized activity recognition. Publications are grouped according to the research field in which the work was conducted.

As early as 1999, the project *TEA* [17] proposed methods for recognizing context with low-level sensors, demonstrating the general feasibility on an extended Nokia mobile phone. The context-aware mobile phone *SenSay* [18] is designed to, among other features, automatically turning the ringer on and off and prevent inaudible ringer volume in a loud environment. However it requires an external sensor box on the user's hip as well as ambient microphones on their body. Another approach using special wearable sensors is the *Mobile Sensing Platform* (MSP) [7], which was specifically developed for activity recognition. A specific activity recognition system built on the MSP is *UbiFit* [8]. Wearable sensors – those of the *Nike+iPod Sport Kit* – are also employed in the *iLearn* system [16] for Apple's *iPhone*.

An example of research in the field of artificial intelligence is [15], in which a triaxial accelerometer was used to detect eight activity classes. Activity recognition based on fuzzy classifiers is presented in [11][21]. [11] uses a Fuzzy Inference System (FIS) and in [21] neuro-fuzzy classifiers are used. In [11] a recognition rate of 95% for four classes and three subjects could be reached, where in [21] eight classes for seven subjects can be recognized with 93% accuracy. Both use external sensors with a fixed body position and do not do the recognition on an embedded device.

In [12] an activity recognition module is applied to a health care monitoring system. User activities of daily living are detected via two accelerometers, firmly positioned at the waist and above the knee. The recognition rates for 15 subjects and 5 classes for the acceleration sensors are 95% accuracy using a decision tree. A smart buckle is used in [6] for activity recognition in medical monitoring applications based on a 3-axis accelerometer sensor in combination with an optical sensor in a fixed position. Conclusive information about the over-

Research field	Health Care				Pervasive		Other			Mobile					SPAR
Publication	[12]	[6]	[13]	[19]	[14]	[2]	[11]	[21]	[15]	[4]	[10]	[16]	[18]	[8, 7]	(this)
# recognized classes	5	9	8	6(9)	6	20	4	8	8	6	6	4	4	5	10
Overall recog. rates (%)	94	-	85	96	80	73(84)	95	93	73(91)	80	82	97	-	77(94)	up to 97
# subjects	15	-	6	-	6	20	3	7	2	-	3	8	-	12	20
# acc. sensors, **i**n-/**e**xt.	2 (e)	1 (e)	1 (e)	1 (e)	1 (i)	5 (e)	1 (e)	1 (e)	1 (e)	1 (i)	3 (e)	2 (i/e)	2 (e)	1 (e)	2 (i)
Sens. pos.: **f**ix/**n**on-fix	f	f	f	f	n	f	f	f	f	n	f	f/n	f	f	n
Class.: **c**entr./on **d**evice	c	c	c	c	d	c	c	c	c	-	d	d	c	d	d

Table 1: Characterized representations of personal acivity recognition approaches

http://www.digibib.tu-bs.de/?docid=00033585 31/05/2010

all recognition rates could not be found in the paper. A triaxial accelerometer was used in combination with a heart rate sensor in [13], located firmly on the dominant thigh. For the six test subjects an average recognition rate of 85% for eight classes was achieved. Another publication utilizing an accelerometer in a healthcare setting is [19]. An accuracy of 96% could be achieved for 6 classes with a sensor fastened to the user's wrist.

[2] is a highly cited publication on activity recognition in pervasive computing using acceleration sensors located on four limb positions and the hip. In [14] the *eWatch* sensing platform is used to detect six activities. The *eWatch* is carried in the test subjects pocket among other body positions. For six users a recognition accuracy of 80% was reached. [4] and [10] both do activity recognition for mobile phones, where [4] only uses sensors and resources native to the phone for sensory acquisition and classification, but just six activities are distinguished with only 80% recognition accuracy on average. In [10] a wristband in combination with a mobile phone is used to recognize six activity classes.

While today many activity recognition systems exist, few of them combine the following desirable features: The system should be able to achieve high overall recognition rates for a reasonable amount of activity classes. For that task it should only employ the internal sensors of an unmodified commodity cellphone on which the recognition is done internally. Furthermore the system should have been evaluated using a fairly high number of test subjects.

Of the systems displayed in table 1, only one system [2] has been designed to recognize more classes than the system presented in the paper at hand. It also has been evaluated using the same high number of subjects. However, the recognition rates of said system are significantly lower than those of ours, in spite of the fact that it uses five external acceleration sensors that need to be placed in fixed positions. In terms of high recognition rates, SPAR ties with *iLearn* [16], whereas it goes without the use of external fixed sensors. Of the systems that were surveyed, only ours and two others [14, 4] exclusively employ non-fixed sensors. Again, the recognition rates lie well below the ones our system achieves. One of these systems [14] also does the classification on the device, as well as SPAR and three others [10, 16, 8]. However, the latter three employ fixed sensors for the activity recognition. When observing table 1, SPAR is the only system combining the desired features described above. In addition, SPAR has the unique features of being service-oriented and personalizable to the individual as well as the community as a whole.

2 The Architecture

The problem of personalized activity recognition was solved with a service based approach. The service consists of several steps of activity classifier training, optimal classifier selection, classifier personalization, and user accuracy feedback. Each step is realized by different components either located on a server or the user's phone. To begin, the user has to collect a small amount of activity data which the service uses to select an initial recognizer that is delivered to the user device. In the next step the recognizer is personalized, improving recognition rates. This step requires more time for calculation, but is still very responsive as will be seen later. The data provided by the user is used in a recurring training process, further improving recognition. Based on the user feedback, the server processes can further improve the activity recognition for the specific user as well as the global accuracy of the service for the community as a whole. Each step can be repeated at any time and certain services further improving activity recognition are constantly running, making it possible to achieve recognition rates near 100% within the community. All steps and components of SPAR are displayed in Fig. 2.

Activity Classification Module Set (ACMS): The ACMS is the collection of classification modules which are running on the user's mobile phone at a given time.
Global Trainer Service (GTS): The GTS is the key component which trains new Activity Classification Module Sets (ACMS). The GTS frequently checks the database for new user activity data sets or for combinations of user data which have not been used. When data is found, the GTS creates a new ACMS using the new data combination. Through the GTS the database is constantly filled with new ACMSs, where badly performing ACMSs are replaced to ensure that the database always consists of the best performing ACMSs.
Personal Trainer Service (PTS): The PTS selects which activity classification modules are delivered to the user

http://www.digibib.tu-bs.de/?docid=00033585 31/05/2010

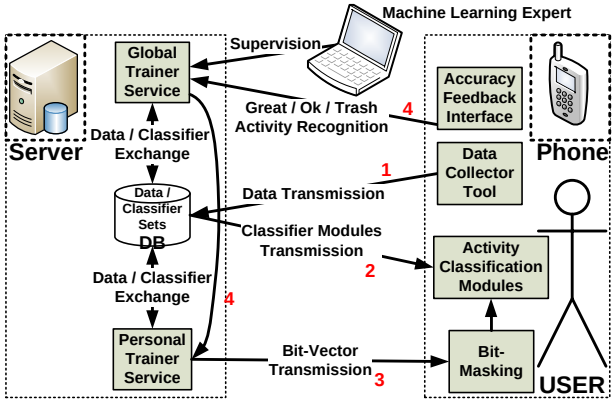

Figure 2: Architecture of activity recognition service.

Figure 3: User feedback over the AFI.

device. The decision is made based on the performance of the classifiers over the annotated data collected by the user. The PTS is also responsible for personalizing the classifier modules; a process accomplished using a genetic algorithm to enable and disable features of the training data via bit-masking. This process will be explained in depth further on in this paper.

Data/Classifier Set Database: This unit is the central Storage for the global ACMS and user data.

Bit-Masking: The bit-masking provides a method for personalizing an ACMS without destroying its original activity recognition capabilities.

Data Collector Tool (DCT): The DCT on the mobile phone collects annotated activity data. The user selects the activity they want to perform from an extensible set of activities in a drop-down-list, carries out the activity, and then pushes a button to stop recording.

Accuracy Feedback Interface (AFI): With the AFI, the user can give feedback to the GTS/PTS as to whether their activity recognition is working. Three kinds of feedback messages are provided, recognition was *well done*, *ok*, or should be thrown into the *trash can* (Fig. 3). This feedback is not used for personalization per se, as the personalization process is done using the bit-mask, but rather could be used to change the classification process in general, though this is not currently implemented and outside of the scope of this paper.

2.1 SPAR Workflow - How the Service Works

First the user downloads the SPAR components to the phone. Second, a small amount of initial data must be collected, where each of the activities available in the drop down list of the DCT is done for 1-2 minutes. When all of the activities are complete the data is transmitted (step 1 Fig.2) to the SPAR server. At this point the interaction between the user and the service is finished and will only be re-initiated over the AFI if necessary. On the server the PTS selects the best performing ACMS from the database based on the initial data and transmits the set to the user device (step 2 Fig.2). This step has nearly no delay, since only a search over the pre-existing ACMSs must be done and no training is performed. The transmitted ACMS can now run on the users phone and recognize activities with an initial accuracy. Meanwhile the PTS is training the personalization of the ACMS currently running. This process takes time (depending on efficiency and complexity about 1 hour), but since an ACMS is already running on the user's phone, the delay is not directly recognizable for the user. The personalization data, which is just a bit-vector and not a complete ACMS, is transmitted to the phone in step 3 (Fig.2). On the phone the bit-masking component personalizes the ACMS, which can be done during runtime in between classifications. Now the phone

http://www.digibib.tu-bs.de/?docid=00033585 31/05/2010

should have reasonable activity recognition rates (see evaluation), but the process runs further on the SPAR server. The GTS constantly trains new combinations of ACMSs, in which the new user's data is included as well. Over time, an ACMS is present in the database which has been trained on the data from the new user and can be transmitted to the user's phone (recognition rates can rise to above 97%). This new ACMS can be personalized again via the PTS and therefore be further improved. The user can also give feedback to the system via the AFI (step 4 Fig.2), but this is not necessary. If the SPAR is in a faulty state due to bad user data, an expert can intervene.

3 Activity Classification Module Set

For activity recognition we use a novel Activity Classification Module Set (ACMS), through which we are able to classify a large number of classes with reduced calculation effort. The ACMS system can provide accuracy of over 97%. In this section we first explain a simplified monolithic activity classification approach and then extend the architecture to the ACMS.

3.1 Recurrent Activity Classification

The classification consists of several steps of processing a real world value to a tuple of the class recognized and a fuzzy uncertainty value (Fig. 4). In the first step, sensors convert the real world signal into a digital measurement. Next, the desired features are extracted from the measurements. In the third step, a Recurrent Fuzzy Inference System (RFIS) maps the features onto a classifiable linear set. The outcome of the mapping is fed back. The linear set is fuzzily classified according to designated fuzzy numbers in the last step.

1. Feature Extraction: For the application in this paper, we used acceleration sensors for activity recognition. We segment the sensor streams into 80 millisecond windows for each of the three axis of the two acceleration sensors. This frame length results in 8 samples (∼100Hz sampling rate) per frame. The features used for activity recognition with acceleration measurements are mostly variance and mean values, since they can be calculated with low resource consumption and give good

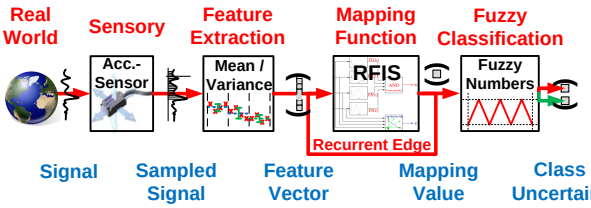

Figure 4: Online system architecture for classification and fuzzy uncertainty.

results. These features were used to preprocess the accelerometer data, the two 3-axis accelerometer sensors lead to a 12-dimensional feature vector $\overrightarrow{v}_t$ at time t.

2. Recurrent FIS Mapping: Takagi, Sugeno and Kang [20] (TSK-) FISs are fuzzy rule-based structures, which are especially suited for automated construction. In TSK-FIS, the consequence of the implication is not a functional membership to a fuzzy set, but a constant or linear function. In our case we use the linear functional consequences f_j, which are weighted with the respective input membership function μ_j in the overall output of the TSK-FIS. This output will be assigned to a tuple of class and fuzziness, which is described in the *Fuzzy Classification* paragraph. The consequence f_j of the rule j depends on the input vector $\overrightarrow{v}_t$ at time t of the TSK-FIS:

$$f_j(\overrightarrow{v}_t) \quad := \quad a_{1j}v_1 + .. + a_{nj}v_n + a_{(n+1)j}$$

Since we deal with highly correlated features, especially when accelerometers are used, we employ rules with a single covariant antecedent:

$$\text{IF } \mu_j(\overrightarrow{v}_t) \text{ THEN } f_j(\overrightarrow{v}_t) \tag{1}$$

$$\mu_j(\overrightarrow{v}_t) := e^{-\frac{1}{2}(\overrightarrow{v}_t - \overrightarrow{\mathbf{m}}_j)\Sigma_j^{-1}(\overrightarrow{v}_t - \overrightarrow{\mathbf{m}}_j)^T} \tag{2}$$

The antecedent membership function $\mu_j(\overrightarrow{v}_t)$ is multiplied with the consequence $f_j(\overrightarrow{v}_t)$, after which the sum over all rules j is divided by the sum of all membership functions $\mu_j(\overrightarrow{v}_t)$. The resulting formula for the covariant TSK-FIS is defined as follows:

$$\mathbf{S}(\overrightarrow{v}_t) := \frac{\sum_{j=1}^{m} \mu_j(\overrightarrow{v}_t) f_j(\overrightarrow{v}_t)}{\sum_{j=1}^{m} \mu_j(\overrightarrow{v}_t)} \tag{3}$$

The outcome of the mapping at time t is fed back as input dimension n for the TSK-FIS mapping at $t+1$.

http://www.digibib.tu-bs.de/?docid=00033585 31/05/2010

The recurrence not only delivers the desired uncertainty level, but also stabilizes and improves the mapping accuracy. Instead of 'Recurrent TSK-FIS' we use the simpler term RFIS in the remainder of this paper. For more detailed information on this process please refer to [20].
3. Fuzzy Classification: The output of the RFIS $\mathbf{S}(\overrightarrow{v}_t)$ at time t is the normalized weighted sum of the functions $f_j(\overrightarrow{v}_t)$ of the rules j. The returned values numerically encode the classes. The assignment of the RFIS mapping result $\mathbf{S}(\overrightarrow{v}_t)$ to a class is done fuzzily, so the result is not only a class identifier, but also a membership, representing the reliability of the classification process. Each class c_k is interpreted by a set of a triangularly shaped fuzzy numbers [1] (eqn. 4):

$$\mu_{c_k}(x) = \begin{cases} \max(0, 1 - 2 \cdot (c_k - x)) & \text{, when } x \leq c_k \\ \max(0, 1 - 2 \cdot (x - c_k)) & \text{, when } x > c_k \end{cases} \tag{4}$$

The mean of the fuzzy number is the identifier c_k itself. The crisp decision – i.e. which identifier is the mapping outcome – is carried out based on the highest degree of membership to one of the fuzzy numbers in the classification $\mathbf{K}$ (eqn. 5).

$$\kappa(x) = \begin{cases} (c_1, \mu_{c_1}(x)) & \text{, when } \mu_{c_1}(x) = \max_k(\mu_{c_k}(x)) \\ \vdots & \vdots \\ (c_o, \mu_{c_o}(x)) & \text{, when } \mu_{c_o}(x) = \max_k(\mu_{c_k}(x)) \end{cases} \tag{5}$$

The overall output of the classifier module $\mathbf{M}$ (eqn. 6) is a tuple (c_k, μ_{c_k}) of a class identifier and the membership to it, where $c_k \in \mathcal{C}$ and $\mu_{c_k} \in [0, 1]$.

$$\mathbf{M}(\overrightarrow{v}_t) := \mathbf{K}(\mathbf{S}(\overrightarrow{v}_t)) = (c_k, \mu_{c_k}) \tag{6}$$

4. Fuzzy Uncertainty Filter: The classifications vary strongly with respect to fuzziness and therefore in the reliability of the RFIS mapping. Since more classifications are made than needed for most applications, a filter on the fuzzy uncertainty ($\mu_{c_k} \leq \tau$) can improve reliability, but also reduces the number of classifications.

3.2 Activity Classification Module Set (ACMS)

Instead of using one monolithic classifier to classify on all classes $\mathcal{C}$, we use several classifier modules $\mathbf{M}_i : \mathcal{V} \rightarrow \mathcal{C}_i$ (with $i = 1, .., N$) each classifying on a small subset $\mathcal{C}_i \subseteq \mathcal{C}$ of classes. The subsets $\mathcal{C}_i$ are chosen according to the classes $c_{ij} \in \mathcal{C}_i$ semantics, therefore each subset $\mathcal{C}_i$ has its own meta semantic. We call this meta semantic 'conditional context'. To not only recognize the respective classes $c_{ij} \in \mathcal{C}_i$, but also the transition between classifiers $\mathbf{M}_i$, each module yields a complementary class $\overline{c}_i$ as well, where the complementary class represents all classes classified by other modules but not by this one. All classifier modules are chained in a dynamic queue, where the last classifier successfully classifying a class aside from the complementary class is moved to the front. The idea behind this re-organization of the classifier queue is that modules which are successive in the queue, are successive in recognition of input features. Therefore, when the 'active' module recognizes the complementary class $\overline{c}_i$, the whole queue does not need to be evaluated for being able to classify this feature vector $\overrightarrow{v}_t$, but only the next module in the queue yields a positive classification in the optimal case. An example for two states of the classifier modules queue at time $t = 0$ and $t = X$ is displayed in figure 5.

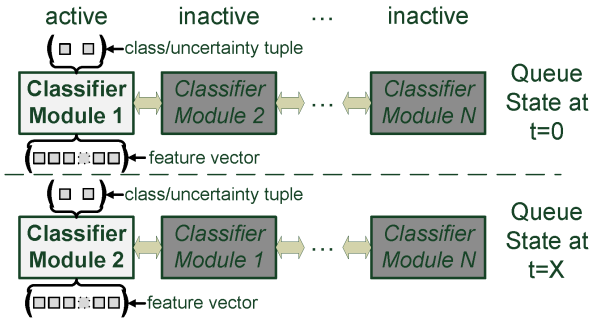

Figure 5: State of modular classifiers in queue at time $t = 0$ and $t = X$

To train these queued classifier modules, we need to train them on the respective classes $c_{ij} \in \mathcal{C}_i$ and on the complementary class $\overline{c}_i$. The training $\mathcal{V}_i^{tr}$ and check data sets $\mathcal{V}_i^{ck}$ for a classifier module $\mathbf{M}_i$ are unified with a selection of input data pairs of all other classifiers $\mathcal{V}_i^{\overline{c}} \subset \bigcup_{k \neq i} \mathcal{V}_k^{tr}$. This selection is labeled zero – which indicates the complementary class $\overline{c}_i$ in every module $\mathbf{M}_i$ – and added to the normal training and check data sets of this classifier. The actual training and check data is therefore $\mathcal{V}_i^{tr \cup \overline{c}}$ and $\mathcal{V}_i^{ck \cup \overline{c}}$, which are called $\mathcal{V}_i^{tr}$ and $\mathcal{V}_i^{ck}$ in the rest of this paper for reasons of simplicity.

http://www.digibib.tu-bs.de/?docid=00033585 31/05/2010

4 Global Trainer Service (GTS)

A classification system which is trained on a large data set of multiple users has disadvantages. First, the training algorithms have long calculation times, since the training data set increases with every user added to it. Second, the resulting Activity Classification Module Set (ACMS) is inaccurate and complex. Since the training data is sizable and the diversity due to the different users is high, the classifier modules need a large number of rules in the RFIS mapping function to map the data. This is because different users have different patterns for certain activities, the data is partly contradictory. In this case, either one user's training data is preferred to solve the conflict, or both are classified with low accuracy.

Instead of training the ACMS on the training data of all users in the data base, we select subsets of user-specific data and train the classifier modules on them. Instead of one set of modules, we end up with various sets trained on subsets of the users. All the ACMSs are stored in a database, where only the best and fittest ACMSs remain and the worse performing are deleted. With each new user added to the database new data combinations of users become possible, so the GTS is constantly running and training new ACMSs.

4.1 Activity Classifier Module Training

To train a set of activity classifier modules, we developed a machine learning algorithm that is fast and requires a minimal amount of supervision from an expert. The part of each activity classification module that needs to be trained is the RFIS mapping function. The recurrence also needs a special machine learning technique, where the recurrent dimension is calculated in every training iteration. The RFIS is completely described by the parameters a_{ij} of the linear functional consequence f_j and the mean vector $\vec{\mathbf{m}}_j$ and the covariance matrix Σ_j of the antecedence membership functions μ_j. These values are identified on an annotated training feature set via a combination of clustering algorithms, linear regression and genetic algorithm generalization. The five step algorithm is displayed in figure 6 and described in the following.

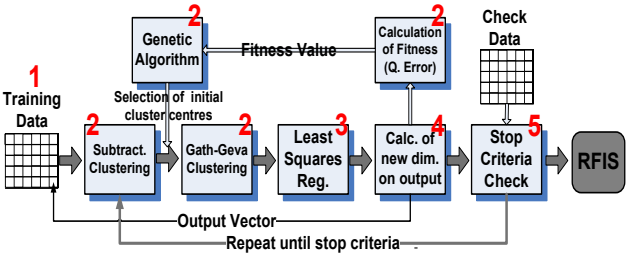

Figure 6: RFIS identification algorithm.

1. Data Annotation and Separation: The training data $\mathcal{V}^{tr}$ is separated according to the class c_j to which the data pairs belong. Clustering on each subset delivers rules that can be assigned to each class. 2. Clustering: Subtractive clustering [5] gives an upper bound for the amount of clusters. Gath-Geva clustering [9] is performed on the set of cluster centers determined through the subtractive clustering. Since the number of clusters for the subtractive clustering is higher than for Gath-Geva clustering (multivariant cluster shapes for covariant sensor data), a genetic algorithm searches for the best subset selection of initial cluster centers for the Gath-Geva clustering. The output of the Gath-Geva clustering is the number of rules m, the mean vector $\vec{\mathbf{m}}_j$ and covariance matrix Σ_j of the membership functions μ_j. 3. Least Squares: Linear regression identifies the parameters a_{ij} of the linear consequence function f_j of the rules $j = 1, .., m$. Minimizing the quadratic error – the quadratic distance between the desired output and the actual output – leads to the solution of an overdetermined linear equation. 4. Recurrent Data Set: The output of the TSK-FIS $\mathbf{S}$ is now calculated over the training data $\mathcal{V}^{tr}$. This output is shifted by one, with a leading zero, and then added to the training data set $\mathcal{V}^{tr}$ as an additional dimension. All data pairs for time $t > 1$ have the output of the FIS mapping of $t - 1$ in the recurrent dimension n. For this data set the steps 1 to 3 are repeated. 5. Stop Criterion: There are two values qualifying for a stop criterion: the mean quadratic error and the classification accuracy. While the mean quadratic error mostly improves the expressiveness of the reliability value rather than the accuracy, optimizing the classification accuracy only improves the percentage of correct classifications. For our evaluation scenario we decided on a fixed number of iterations.

http://www.digibib.tu-bs.de/?docid=00033585 31/05/2010

4.2 Activity Classification Module Set (ACMS) Training

The training $\mathcal{V}^{tr}$ and check data $\mathcal{V}^{ck}$ for one classifier modules set $\mathcal{M}_l = \{\mathbf{M}_{1l}, .., \mathbf{M}_{Nl}\}$ consists of data from randomly selected users ($u_h \in \{u_1, .., u_A\}$), represented by $\mathcal{V}_{u_h}$. The training data $\mathcal{V}^{tr}_{\mathcal{M}_l}$ for the classifier module set $\mathcal{M}_l$ is a subset selection of the data $\mathcal{V}^{tr}_{\mathcal{M}_l} \subseteq \mathcal{V}_{u_g} \cup .. \cup \mathcal{V}_{u_h}$ for $g \neq .. \neq h$, where each of the classes should have equal amount of training data pairs and the data from every user should contribute same amount of data sets. For the check set $\mathcal{V}^{ck}_{\mathcal{M}_l}$ we proceed accordingly.

All the users data is saved in a database, from which different combinations of data are randomly selected. on each selection a new set of classifier modules set $\mathcal{M}_l$ is trained. The set $\mathcal{M}_l$ is then checked with the respective check data $\mathcal{V}^{ck}_{\mathcal{M}_l}$ for classification accuracy. This process is repeated until all combinations of user specific data are trained and checked, where only a certain population of ACMS which have the highest classification accuracy is saved. With new user data coming in through the community, the process can theoretically run indefinitely. The amount of stored classifier module sets should increase with the amount of new user data added to the database. Since storage space is limited, the amount can not be increased indefinitely.

5 Personal Trainer Service (PTS)

The classifier module set $\mathcal{M}_l$ which performs best for the current user is selected and must then be personalized to recognize activities with acceptable accuracy. This is done via a bit-vector masking, which 'activates' or 'deactivates' the input dimensions of the rules of each classifier module's RFIS mapping function. This masking is only temporary, so the original classifier modules still persist and can be reactivated at any time. Furthermore, the bit-vector can be changed and therefore the capabilities of the masked classifier at any time. The bit-vector is specified for the respective user according to the data $\mathcal{V}_{u_h}$ on which the classifier module set $\mathcal{M}_l$ was selected. Since the combinations of bits in the bit-vector exclude a full search, a genetic algorithms is used as a heuristic to limit the search space.

5.1 Activity Classifier Module Set (ACMS) Selection

The first step towards a user's personalized activity classification system is the selection of the best classifier module set from the data base. The selection is done based on an initial data set which the user has previously collected. The data set should consist of a significant amount of data pairs for all activity classes. For every classifier module set $\mathcal{M}_l$ ($l = 1, .., A$) the classification accuracy for the new user data is calculated. The classifier module set $\mathcal{M}_l$ which has the best classification accuracy is then selected for the activity recognition on the user's device. A exemplary classifier module set selection process is illustrated by Fig. 7.

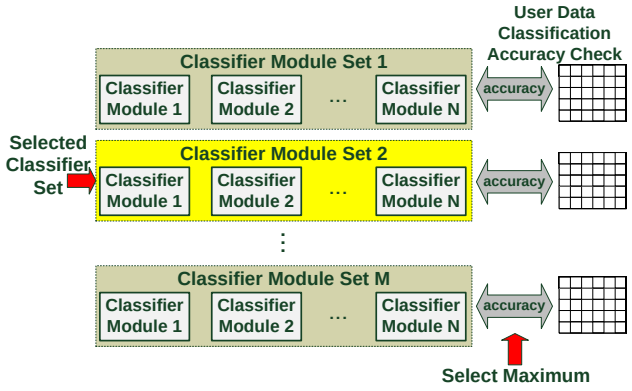

Figure 7: Example for classifier module set selection.

5.2 Bit-Vector for ACMS Masking

The adaption is done via a bit vector, which specifies the 'active' and 'inactive' dimensions of each rule for one module $\mathbf{M}_i$. Therefore the bit vector $bit_{\mathbf{M}_i}$ for module $\mathbf{M}_i$, which has n input dimensions and m_i rules, is $b_i := n \cdot m_i$ positions long. To use the bit vector, an interpretation function $I(\mathbf{M}_i(\overrightarrow{v}_t), bit_{\mathbf{M}_i})$ is defined, that 'switches' the rule's dimensions temporarily without changing the module $\mathbf{M}_i$ permanently. The interpretation function I is defined as a function mapping a module $\mathbf{M} \in \mathbb{F}(\mathbb{R}^n, \mathbb{R})$ together with a bit vector $bit_{\mathbf{M}} \in \{0,1\}^*$ of appropriate length to a module $\mathbf{M}'$:

$$I : \mathbb{F}(\mathbb{R}^n, \mathbb{R}) \times \{0,1\}^* \longrightarrow \mathbb{F}(\mathbb{R}^n, \mathbb{R}) \qquad (7)$$

http://www.digibib.tu-bs.de/?docid=00033585 31/05/2010

Details on the bit vector approach can be found in [3].

5.3 Genetic Algorithm Bit-Vector Specification

A genetic algorithm is used to determine the respective classifier module's bit vector. The space that has to be searched is 2^{b_i}, where $b_i := n \cdot m_i$ is the length of the bit vector $bit_{\mathbf{M}_i}$ for module $\mathbf{M}_i$. A complete search would therefore have a runtime of $O(2^{b_i})$, which is impossible to calculate in a reasonable amount of time. In our experience, the genetic algorithm can find a suboptimal, but appropriate solution in a time span that is acceptable in our application. Nevertheless, we are currently investigating methods to limit the search space. Here, the overlapping of membership functions can indicate which dimensions should be 'deactivated', so overlapping is reduced or even eliminated.

6 Online ACMS Implementation and Evaluation

For the online ACMS evaluation we used the *OpenMoko Neo Freerunner* phone based on the Samsung S3C2442B processor clocked at 266MHz using a Debian Linux operating system enabling rapid prototyping. We are also currently investigating implementations on the iPhone 3G (see video) and the Motorola Milestone running the Android operating system. The ACMS is implemented in C and the parameter setting is provided through a JASON configuration file. The C implementation is divided in different independent code parts: the feature extraction consisting of mean and variance calculations, the RFIS interpretation function, the fuzzy classification and the bit-vector masking.

6.1 Performance Evaluation

To gather meaningful performance data, we must first compute whether the requirements for real-time processing can be achieved: the accelerometer sensor provides 100 samples per second and each classification requires 8 sampled window size in the feature extraction. Thus, if we want to achieve real-time performance, we must be able to perform 12.5 activity classifications per second. Measurements are conducted for three distinct processes: the accelerator data feature extraction, the activity classification and some overhead for the classifier modules switch. For the classification processes, a distinction must be made between the best and worst case: in the best case, the first classifier module of the ACMS would be able to classify the data, and in the worst case, all classifiers modules need to be computed. Each process was run 1000 times to gather significant performance results (See Tab. 2). The results indicate

	best case	worst case (4 modules)
feature extraction	0.012%	0.012%
classification	0.093%	0.38%
total (12.5 classific. per sec.)	1.3%	4.9%

Table 2: Performance measurements.

that we only require 4.9% of the available processing power available. The Neo FreeRunner's processor does not feature floating point instructions, meaning that all floating point calculations have to be simulated using the integer instruction set. With a phone providing a floating point unit (e.g. iPhone 3G) we would need approximately ten times less processing time.

For activity recognition on a mobile phone it is not only necessary to perform in real-time, but also to leave enough processing resources free to be able to execute normal phone applications (e.g. making a call or running navigation software). Processor time for the ACMS was observed for 131 minutes in a trial period in order to evaluate CPU utilization. The measurements showed a CPU usage of 3.3% for the ACMS on average.

7 Offline Activity Classification Evaluation

In order to evaluate the SPAR approach, the system was evaluated offline. First the evaluation settings will be explained, followed by a demonstration of upper and lower bound approximations for the optimal case where the training data for the classifier modules are only gathered from the evaluation user. The lower bound approximation is the accuracy of the classifier modules

http://www.digibib.tu-bs.de/?docid=00033585 31/05/2010

for a data set from the evaluation user, but with a different phone orientation (phone is upside down in the pants pocket). In the next step, an ACMS is selected from the database where data from the current user is not present, meaning there is no ACMS available which was directly trained on data from this user. The accuracy of the best performing ACMS under these circumstances is presented. Next, personalization using the PTS to provide bit-vectors for each module is presented where data from the current user is excluded from the database. Finally the performance of the whole system including data from the evaluation user as well as the community is presented, where different phone orientations and personalization are applied.

7.1 Evaluation Setting

As mentioned before, there is a distinction between context classes and conditional contexts. The classes are the direct output of the classifier, whereas the conditional context is implicitly identified through the classifier module which is currently active. The classes are sorted according to semantics of the conditional context. Three general conditional contexts were determined, the phone is on a table, the phone is in the trouser pocket and the user has the phone in her hand. Since the amount of classes (6 classes) for the "phone in trouser pocket" conditionality is too much to be classified with one classifier module, we decided to use two modules for this state, each classifying onto three classes. These two subset classifiers do not have mutually exclusive classes, but we decided to sort the classes into subsets according to the amount of movement in the acceleration patterns. The combinations of contextual states, classes and classifiers for the acceleration data classification are shown in Tab. 3.

We collected data from 20 users (16 male, 4 female), aged 20 to 32. All users had experience with the use of mobile phones. We tried to ensure that the collection of activities was as realistic as possible. Each test user generated 2-3 minutes of data for each of the ten activity classes during normal everyday activities, resulting in over 500 minutes of data in total.

The classifier module sets are trained on randomly selected sets of four users. 19 of the 20 users were used during the training phase, leaving the data from one user for evaluation of the SPAR system. The male test subject used for the evaluation provided data with two different orientations for the 'pocket' activities. All the test data used in the following evaluation is randomized in slices of 16 data pairs. This is a stress test in the evaluation, since due to the recurrence in the classifier modules, the most false-positives happen in between activity classes.

Conditional Context	Context Class	Class No.	Classifier Module
Phone in users trouser pocket: *no movement*	user is sitting	1	$\mathbf{M}_1$
	user is standing	2	
	user is lying	3	
movement	user is walking	4	$\mathbf{M}_2$
	user is climbing stairs	5	
	user is cycling	6	
Phone on table:	no movement	7	$\mathbf{M}_3$
Phone in users hand:	just holding	8	$\mathbf{M}_4$
	talking on phone	9	
	typing text message	10	

Table 3: Conditional contexts, classes and classifier modules for the acceleration sensor.

In the following tables the recognized activity classes are filtered on a threshold of $\tau = 75$. This reduces the amount of classes which are recognized by a certain amount, depending on the general detectability of the respective activity. Which percentage of the classification remain after the filtering is shown in the last row of the confusion matrices. We expect that a remainder of more than 8% of classifications to be acceptable, so on average still one classification passes through the reliability filter per second.

7.2 ACMS Trained on Data of Evaluation User

The best results for activity recognition can be achieved when the ACMS is directly trained on the current user. This gives us an upper limit for activity recognition accuracy for a 3-minute training cycle. Data collection resulted in about 2250 feature vectors per class and 22500 in total for all ten classes. Out of this annotated data set, 600 samples per class were extracted for training data $\mathcal{V}^{tr}$ and 400 for check data $\mathcal{V}^{ck}$. For training the complementary class of each of the modules 1200 data pairs were randomly selected from training data of the other classifier modules and added to the training data. For the check of the complementary class 800 data pairs

http://www.digibib.tu-bs.de/?docid=00033585 31/05/2010

were added to the check data. The remaining data - ∼ 1250 annotated feature vectors per class - was used to test the ACMS. The upper limit results for the trained ACMS are presented in table 4.

	$\mathbf{M}_1$ (98.1%)			$\mathbf{M}_2$ (96.4%)			$\mathbf{M}_3$	$\mathbf{M}_4$ (99.3%)		
	1	2	3	4	5	6	7	8	9	10
1	97.8	0.8	0.1	4.2	0.2	0.7	0.4	0.0	0.3	0.0
2	0.6	96.3	1.2	0.9	0.0	0.0	0.0	0.0	0.1	0.0
3	0.0	0.0	97.4	0.0	0.1	0.0	0.0	0.0	0.0	0.0
4	0.8	1.3	1.2	92.3	0.1	0.2	0.0	0.9	0.6	0.0
5	0.0	0.0	0.0	0.3	97.7	1.6	0.0	0.0	0.0	0.0
6	0.0	0.0	0.1	0.0	0.0	97.1	0.0	0.0	0.0	0.0
7	0.5	0.0	0.0	0.0	0.0	0.0	99.5	0.0	0.0	0.0
8	0.3	1.0	0.0	2.2	1.5	0.3	0.1	97.6	1.1	0.0
9	0.1	0.5	0.0	0.2	0.4	0.1	0.0	1.4	97.8	0.8
10	0.0	0.0	0.0	0.0	0.1	0.0	0.0	0.1	0.1	99.1
	83.9	83.2	93.5	69.4	57.5	74.5	91.7	86.3	83.2	93.5

Table 4: Upper limit classifier system, solely trained on the evaluation user. Filter threshold: $\tau = 0.75$. Recognition rate conditional context: 98.3%. Overall rate: 97.3%.

A lower baseline limit can be expressed when the classifier was trained on one phone orientation (e.g. for the phone in the pants pocket) and the evaluation is carried out on data from the user when carrying it in a different position, which represents a major problem in activity recognition. The recognition results for the test data with a different phone orientation then the training data are shown in figure 5. Here the classes for the conditional context 'phone in user's trousers pocket' have extremely low recognition rates, where the activities with only one possible orientation still have high accuracy. We will show in the evaluation of our service-based approach that the recognition accuracy achieved is not significantly lower than the upper limit and can exceed accuracy for the lower baseline limit.

7.3 Evaluation User Excluded from Training Data for ACMS

As explained before, the upper limit classifier modules cannot be supported in a community based approach due to several reasons. We will also show that we can exceed the lower baseline limit for a different phone orientation with our service based approach. We will present example results for each of our service's steps.

	$\mathbf{M}_1$ (64.6%)			$\mathbf{M}_2$ (51.4%)			$\mathbf{M}_3$	$\mathbf{M}_4$ (99.7%)		
	1	2	3	4	5	6	7	8	9	10
1	20.7	44.7	0.1	30.1	10.2	20.0	0.1	0.0	0.1	0.0
2	0.2	29.3	0.7	12.4	2.5	21.8	0.0	0.0	0.1	0.0
3	0.0	0.0	98.2	1.0	0.2	0.0	0.0	0.0	0.0	0.0
4	0.0	20.9	0.5	55.2	84.1	13.5	0.0	0.2	0.5	0.0
5	0.0	0.6	0.2	0.6	0.6	0.3	0.0	0.0	0.0	0.0
6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
7	0.0	0.0	0.0	0.0	0.3	0.3	99.8	0.0	0.0	0.0
8	79.2	4.1	0.3	0.7	2.0	40.6	0.1	98.5	0.7	0.1
9	0.0	0.4	0.0	0.0	0.1	3.5	0.0	1.3	98.5	0.6
10	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.1	99.3
	39.6	22.8	92.1	31.4	34.8	11.9	94.6	89.8	89.8	96.2

Table 5: Lower limit classifier system for position two, which was solely trained on the actual user's position one. Filter threshold: $\tau = 0.75$. Recognition rate conditional context 78.9%. Overall recognition rate: 60.0%.

7.3.1 GTS Trained ACMS for Random Selection of User Data

The Global Trainer Service (GTS) has trained 20 Activity Classification Module Sets for randomly selected sets of four users from the 19 users in the database. The training data $\mathcal{V}^{tr}$ consisted of 300 randomly selected feature vectors per class and user, which makes 1200 pairs per class. Again, training data from the other modules was added to train the respective classifier module on the complementary class. In total we used 12000 data pairs for training of all four classifier modules. The check data was 6000 data pairs in size. The classification accuracy for the trained ACMS on the train $\mathcal{V}^{tr}$ and check data $\mathcal{V}^{ck}$ are shown in table 6. For some of

User Combination	Accuracy (%) $\mathcal{V}^{tr}$	$\mathcal{V}^{ck}$	User Combination	Accuracy (%) $\mathcal{V}^{tr}$	$\mathcal{V}^{ck}$
17, 2, 1, 11	86.0	83.1	18, 4, 7, 15	85.6	81.4
12, 11, 1, 7	83.9	83.2	4, 7, 2, 15	85.06	83.7
6, 14, 15, 16	NaN	NaN	12, 17, 7, 16	NaN	NaN
16, 13, 12, 7	NaN	NaN	9, 13, 18, 17	82.9	82.5
3, 6, 8, 4	85.2	82.7	13, 6, 2, 16	82.7	80.2
15, 16, 12, 2	66.3	66.1	3, 17, 16 14	69.5	64.5
13, 8, 6, 1	84.5	80.2	13, 1, 16, 2	83.9	80.7
10, 7, 12, 2	84.7	81.2	10, 15, 4, 5	84.7	82.8
5, 1, 10, 1	72.5	71.4	7, 5, 17, 12	82.7	80.8
9, 5, 18, 8	82.7	69.0	5, 13, 19, 8	NaN	NaN

Table 6: GTS combinations of user data and accuracy of trained ACMS on training and check data. Gray rows indicate faulty ACMSs that are deleted from the database.

http://www.digibib.tu-bs.de/?docid=00033585 31/05/2010

the ACMS an error occurred during training and the accuracy on the training and check data is 'Not a Number (NaN)'. These ACMS are deleted by the GTS from the database along with badly performing ACMSs.

7.3.2 Best Selection of Classifiers

A new user now demands an ACMS to recognize the 10 activity classes in our evaluation setting. The PTS selects the best performing ACMS on the user data and uploads the ACMS to the user phone. The best selection is the ACMS trained on the users 10, 7, 12, and 2. The recognition rate without filtering is 56%. The confusion matrix for a filter threshold of $\tau = 0.75$ is shown in figure. 7.

	$\mathbf{M}_1$ (7.7%)			$\mathbf{M}_2$ (79.7%)			$\mathbf{M}_3$	$\mathbf{M}_4$ (98.5%)		
	1	2	3	4	5	6	7	8	9	10
1	0.7	1.1	0.0	7.8	0.9	1.7	0.2	0.2	0.1	0.0
2	0.5	4.6	14.8	0.0	0.3	0.6	0.1	0.0	0.0	0.0
3	0.0	0.2	1.1	0.0	0.0	0.0	0.1	0.0	0.0	0.0
4	62.6	1.9	71.6	44.1	0.9	1.1	0.7	0.9	3.0	0.4
5	1.4	0.2	2.3	0.6	95.8	5.8	0.1	0.0	0.0	0.0
6	33.6	0.0	0.0	0.0	0.6	90.3	0.0	0.0	0.0	0.0
7	0.0	0.0	0.0	0.0	0.0	0.0	98.9	0.0	0.0	0.0
8	1.1	3.5	10.2	38.5	1.5	0.5	0.0	97.8	1.8	0.5
9	0.0	88.5	0.0	9.0	0.0	0.0	0.0	1.1	95.1	3.8
10	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	95.3
	14.1	37.9	5.6	19.3	12.9	31.3	46.5	20.6	46.1	36.2

Table 7: Best performing selection of classifiers (trained on user w, x, y and z). Filter threshold: $\tau = 0.75$. Recognition rate conditional context 71.2%. Overall recognition rate: 62.4%.

A mean recognition rate of 62.4% is too low, but if we look closer, one can see, that the more user invariant activity classes were recognized with over 90% accuracy. Also the conditional contexts have an accuracy in detection of 71.2%, which is already a good recognition rate, for an ACMS that was provided without any delay.

7.3.3 Personalization Data and Test Data Have Same Orientation

The PTS now provides a bit-vector for masking the selected ACMS, and therefore personalizes the activity recognition. The resulting confusion matrix is displayed in figure 8. The recognition rates have improved significantly by $23.6PP$ up to 86%. Still, classes no. 4 and no. 6 have low recognition rates. These classes are mostly misclassified onto class no. 1 of a different classifier module. Here the personalization process of the PTS can be improved, so that the classifier module $\mathbf{M}_1$ is not personalized to the point it interrupts the capabilities of module $\mathbf{M}_2$. But this is a trade of, where the stop criterion of the genetic algorithm used in the PTS plays the key role. Either modules have equal average recognition rates or one is favored in the disadvantage of another one. This can also happen between classes classified through the same module.

	$\mathbf{M}_1$ (90.2%)			$\mathbf{M}_2$ (72.8%)			$\mathbf{M}_3$	$\mathbf{M}_4$ (97.0%)		
	1	2	3	4	5	6	7	8	9	10
1	85.4	2.7	1.0	31.5	1.7	29.8	0.2	0.2	5.9	0.9
2	1.4	84.9	0.4	1.5	0.4	0.9	0.0	0.0	0.8	0.2
3	0.0	0.7	94.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	1.3	9.1	2.4	54.9	0.4	0.0	0.1	0.0	0.8	0.0
5	0.1	1.0	1.3	0.0	92.5	5.5	0.1	0.0	0.0	0.1
6	11.6	0.0	0.0	0.0	3.9	61.3	0.0	0.0	0.0	0.0
7	0.0	0.0	0.0	0.0	0.0	0.0	99.3	0.0	0.0	0.0
8	0.0	1.6	0.1	10.5	1.0	1.7	0.2	98.4	1.2	0.0
9	0.0	0.0	0.4	1.5	0.0	0.9	0.1	1.4	90.9	0.9
10	0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.4	97.9
	19.9	24.9	42.5	23.4	18.4	11.6	41.9	22.6	25.9	40.8

Table 8: Best performing selection personalized on new user. Filter threshold: $\tau = 0.75$. Recognition rate conditional context: 89.8%. Overall rate: 86.0%.

7.3.4 Personalized Opposite Orientation in Training and Test Data

The PTS has personalized the ACMS via a bit-vector masking according to a data set, where the phone has only one orientation in the user's trouser pocket. We are investigating now what happens if the user is carrying the phone with a different orientation, which is a situation that frequently occurs in daily usage of mobile phones. The results of a test set with about 22000 data pairs for a different phone orientation are shown in table 9. The accuracy is on average low at 63.6%, but as mentioned before, we need to compare these results with the lower baseline approach. There the ACMS was directly trained on the users data with one phone orientation, which is the general approach of a user who has no knowledge of machine learning techniques. Compared to the lower baseline, we can exceed the recognition rate by $3.6PP$. With the recognition rate for the conditional contexts we can reach a more significant improvement

http://www.digibib.tu-bs.de/?docid=00033585 31/05/2010

	$\mathbf{M}_1$ (79.3%)			$\mathbf{M}_2$ (83.6%)			$\mathbf{M}_3$	$\mathbf{M}_4$ (99.4%)		
	1	2	3	4	5	6	7	8	9	10
1	54.4	87.8	0.6	1.1	0.8	9.0	0.0	0.0	1.4	0.1
2	1.8	1.5	0.6	0.2	0.1	0.1	0.0	0.0	0.2	0.0
3	0.0	0.0	91.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	14.0	5.9	2.3	85.3	77.5	73.3	0.0	0.0	0.0	0.1
5	3.5	0.0	1.0	5.8	8.7	0.1	0.0	0.0	0.0	0.0
6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
7	0.0	0.0	0.0	0.0	0.0	0.0	99.9	0.0	0.0	0.0
8	22.8	4.2	1.0	7.4	12.3	13.6	0.0	99.4	1.2	0.0
9	0.0	0.4	3.3	0.1	0.7	3.8	0.0	0.6	96.1	0.5
10	0.0	0.2	0.1	0.0	0.0	0.1	0.0	0.0	1.1	99.3
	1.2	15.3	43.1	43.5	48.0	44.5	94.4	50.0	59.2	82.3

Table 9: Best performing selection personalized on new user, test data has different orientation as data the classifiers personalized on. Filter threshold: $\tau = 0.75$. Recognition rate conditional context 90.6%. Overall recognition rate: 63.6%.

by 11.7*PP* up to 90.6%. Here user feedback through the AFI can trigger a repeated personalization via the PTS or the GTS could have already trained a new ACMS which would perform better with the different phone orientation. The various possibilities of improving the ACMS is the strength of SPAR.

7.4 Evaluation Results Summary and Discussion

Since our Service-based Personal Activity Recognition (SPAR) consists of many steps, we now want to summarize and discuss the results. The recognition accuracy for our example implementation compared to upper and lower baseline is shown in table 10. The numbers show,

Service Step	Accuracy of Activity Recognition	Accuracy of Cond. Context Recognition	Delay Until Available
Upper Limit	97.3%	98.3%	days
Lower Limit	60.0%	78.9%	none
Selected ACMS	62.4%	71.2%	seconds
Pers. ACMS	86.0%	89.8%	hours
Pers. ACMS tested with diff. orientation	63.6%	90.6%	hours

Table 10: Summary of recognition accuracy results.

that we can reach reasonable recognition rates (71% for conditional contexts) and high rates (> 90%) for some classes using initial data with SPAR. This activity recognition is delivered through our service architecture with a delay of only seconds. The personalization step does require a longer computation period (in range of hours), where the key roles for duration are the complexity of the ACMS and the efficiency of implementation of the PTS. After personalization we reach recognition rates of up to 90%, where only two classes have low accuracy which can be changed through a better tradeoff in the PTS algorithms. With the SPAR we also can exceed (11.7*PP* for cond. context) the lower baseline limit for opposite phone orientations not included in the GTS training. In general, with the GTS constantly training new combinations of ACMS on the user data in the database, we can reach the upper limit of over 97% accuracy with a runtime duration of days. Due to the random selection of training data, the delay can also only be minutes.

8 Conclusions

We presented a Service-based Personal Activity Recognition (SPAR) which aims to support activity recognition on mobile phones for common users. No user knowledge of machine learning is needed, nor are behavioral changes required in order to achieve results. The service provides an interface for annotating initial data, which then suffices for selection of the proper classification modules. The services running on the server constantly improves recognition through personalization and simultaneously optimizes recognition throughout the community of users committed to the SPAR. We presented evaluation for each of the service step and compared the results to upper and lower limits. The results indicate that SPAR produces recognition rates of over 97% for the individual user as a part of a community, while also improving results for users carrying their mobile devices in uncommon orientations. Furthermore, the results show that SPAR has a low impact profile on processing usage as well as power consumption.

References

[1] O. A. AbuAarqob, N. T. Shawagfeh, and O. A. AbuGhneim. Functions defined on fuzzy real numbers according to zadehs extension. *International Mathematical Forum*, 3(16):763 – 776, 2008.

http://www.digibib.tu-bs.de/?docid=00033585 31/05/2010

[2] L. Bao and S. S. Intille. Activity recognition from user-annotated acceleration data. *PERVASIVE*, 2004.

[3] M. Berchtold, T. Riedel, K. van Laerhoven, and C. Decker. Gath-geva specification and genetic generalization of tsk fuzzy models. *SMC08*, 2008.

[4] T. Brezmes, J.-L. Gorricho, and J. Cotrina. Activity recognition from accelerometer data on a mobile phone. In *Proceedings of the IWANN '09*, pages 796–799. Springer, 2009.

[5] S. Chiu. Method and software for extracting fuzzy classification rules by subtractive clustering. *IEEE Control Systems Magazine, pp. 461-465*, 1996.

[6] Y. Cho, Y. Nam, Y.-J. Choi, and W.-D. Cho. Smartbuckle: Human activity recognition using a 3-axis accelerometer and a wearable camera. In *HealthNet '08*, pages 1–3, New York, 2008. ACM.

[7] T. Choudhury, G. Borriello, and S. Consolvo, et al. The mobile sensing platform: An embedded activity recognition system. *IEEE Pervasive Computing*, 7(2):32–41, 2008.

[8] S. Consolvo, D. W. McDonald, and T. Toscos, et al. Activity sensing in the wild: a field trial of ubifit garden. In M. Czerwinski, A. M. Lund, and D. S. Tan, editors, *CHI*, pages 1797–1806. ACM, 2008.

[9] I. Gath and A. B. Geva. Unsupervised optimal fuzzy clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence, vol 11(7), pp 773-781*, 1989.

[10] N. Györbíró, A. Fábián, and G. Hományi. An activity recognition system for mobile phones. *Mobile Networks and Applications*, 14(1):82–91, 2009.

[11] M. Helmi and S. AlModarresi. Human activity recognition using a fuzzy inference system. *FUZZ-IEEE*, pages 1897 – 1902, 2009.

[12] Y.-J. Hong, I.-J. Kim, S. C. Ahn, and H.-G. Kim. Mobile health monitoring system based on activity recognition using accelerometer. *Simulation Modelling Practice and Theory*, 18(4):446 – 455, 2010.

[13] D. Maguire and R. Frisby. Comparison of feature classification algorithm for activity recognition based on accelerometer and heart rate data. *9th. IT & T Conference*, 2009.

[14] U. Maurer, A. Smailagic, D. P. Siewiorek, and M. Deisher. Activity recognition and monitoring using multiple sensors on different body positions. In *BSN '06*, pages 113–116. IEEE Computer Society, 2006.

[15] N. Ravi, N. Dandekar, P. Mysore, and M. L. Littman. Activity recognition from accelerometer data. In *IAAI*, pages 1541–1546. AAAI Press, 2005.

[16] T. S. Saponas, J. Lester, J. E. Froehlich, J. Fogarty, and J. A. Landay. ilearn on the iphone: Real-time human activity classification on commodity mobile phones. *CSE Technical Report*, 2008.

[17] A. Schmidt, K. A. Aidoo, A. Takaluoma, U. Tuomela, K. V. Laerhoven, and W. V. de Velde. Advanced interaction in context. In *HUC'99*, LNCS, 1999.

[18] D. Siewiorek, A. Smailagic, and J. Furukawa, et al. Sensay: A context-aware mobile phone. In *ISWC '03*, page 248. IEEE Computer Society, 2003.

[19] S. Song, J. Jang, and S. Park. A phone for human activity recognition using triaxial acceleration sensor. *Consumer Electronics, ICCE*, 2008.

[20] T. Tagaki and M. Sugeno. Fuzzy identification of systems and its application to modelling and control. *Syst., Man and Cybernetics*, 1985.

[21] J.-Y. Yang, Y.-P. Chen, G.-Y. Lee, S.-N. Liou, and J.-S. Wang. Activity recognition using one triaxial accelerometer: A neuro-fuzzy classifier with feature reduction. In *ICEC*, pages 395–400. Springer, 2007.